SF 298 MASTER COPY KEEP THIS COPY FOR REPRODUCTION PURPOSES

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	10/30/00	Final--8/1/96-7/31/00

4. TITLE AND SUBTITLE

Prototype for an Open Intelligent Information System

5. FUNDING NUMBERS

DAAH04-96-1-0325

6. AUTHOR(S)

J. S. Deogun, A. Samal, S. Seth

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)

University of Nebraska-Lincoln
303 Admin. Bldg.
Lincoln, NE 68588-0430

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

ARO 36203.3-RT-DPS

11. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The project developed an open, intelligent, distributed information system with a core retrieval engine based on concepts. The research focused on the development of the theory and algorithms for a concept-based retrieval engine.

14. SUBJECT TERMS	15. NUMBER IF PAGES
information systems, information retrieval	18
	16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Prototype for an Open Intelligent Information System

Final Progress Report

Jitender S. Deogun Ashok Samal Sharad C. Seth

October 30, 2000

U.S. ARMY RESEARCH OFFICE

Grant Number: DAAH04-1-0325

Department of Computer Science & Engineering
University of Nebraska-Lincoln

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

1 Statement of the Problem

The goal of this project is to develop a system for an open, intelligent, distributed information system with a core retrieval engine that is based on *concepts*. The system will be implemented at the University of Nebraska-Lincoln. The bulk of the research will be in the development of the theory and algorithms for concept-based retrieval engine. For the browsing tools and user interface design we will rely on the existing technology, but customize it for our purpose.

2 Summary of the most important results

The period covered by this report is August 1, 1996 – July 31, 2000. Most of the objectives were accomplished. From the proposed objectives some objectives were dropped. The most significant decision was to change the design concept. Originally, the design concept based on HTML format but because of evolution in technology it was decided that the design concept should be modified to adapt XML format. Although, this decision slowed the progress but we felt that we must adapt to the new technology.

In Section 2.1, we present the design and development of an Internet information retrieval system using XML/RDF. The research on query enhancement based on user concepts is reported in Section 2.2. The information filtering research based on induction methods is described in Section 2.3. Finally, outcome of research on formal concept analysis is discussed in Section 2.4.

2.1 Internet Information Retrieval Using XML/RDF

New Design of the System

The Internet has emerged as the largest warehouse of information with the introduction of HTML. However, limitations of HTML are becoming increasingly evident in extracting the relevant information quickly and efficiently from the Internet. Therefore, WWW Consortium (W3C) proposed the use of metadata to describe the structural abstractions embedded in HTML documents. Metadata in this context means *data describing Web resources.* The Resource Description Framework (RDF)/Extensible Markup Language (XML) (RDF/XML, for short) model is designed to represent metadata and explore structural information from Web resources. With this in mind a new architecture for the system was developed. The main idea of this system is to perform information retrieval based on the structural information that is hidden in the web. The structural information is expressed by an RDF model and XML syntax. The new system architecture is shown in Figure 1. In the following, we describe the new design and address the implementation issues. This research project develops an approach to improve the precision and recall measures as well as enhancing querying capabilities for Internet IR. The system developed can be divided into three functionalities:

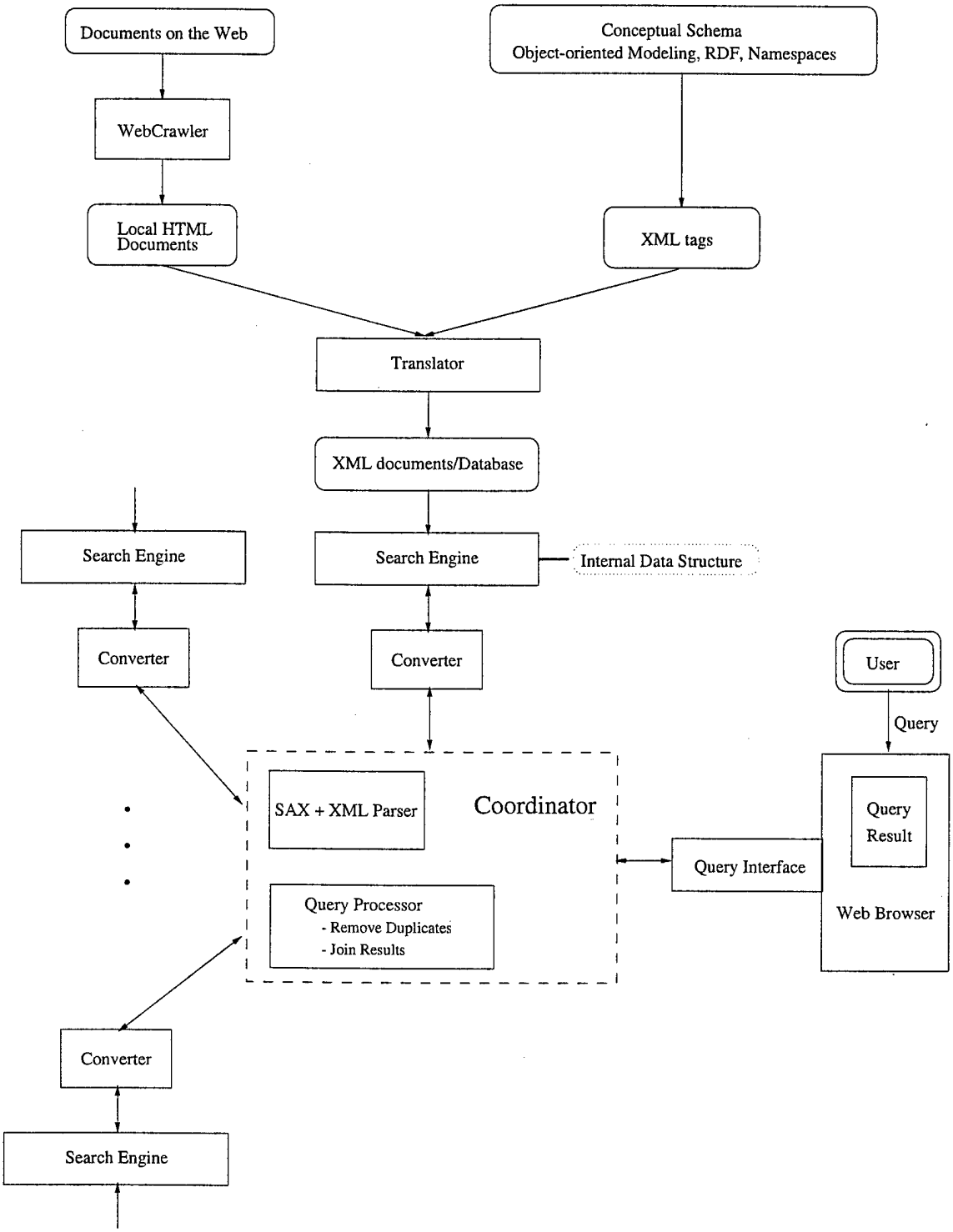

Figure 1: System Outline.

1. Get the HTML hierarchy from Web sites and create XML documents.
2. Validate the XML documents and create an XML Database.
3. Perform the Information Retrieval based on XML Database.

XML-based Information Retrieval

Our XML-based Information Retrieval (IR) model consists of the following processes:

1. Creating an internal representation of XML database.
2. User Queries.
3. IR based on the internal data structure.
4. Buffering and communicating IR results to Query Interface.
5. Query Interface.

Create the Internal Data Structure

Generally there are two ways to perform the IR based on XML database. One is the *pure text-based probing* approach. In this method, when the search engine receives a query, it resorts to full text operations to perform the IR – This process includes locating all possible occurrences of the information by traversing the links and checking, for each occurrence, if it matches the query criteria. This type of *probing* exhibits at least two features; first, it must apply and resort to the Database Schema to do the searching, and the second is that the *probing* is some kind of recurrence of both searching and matching. If the query is an *OR* operation, then probing has to cover all the documents. This approach is usually hard to implement and maintain. Moreover, if the Database Schema is changed, probing yields different results.

The second approach is to create an *Internal Data Structure*. This means the IR process is split into two stages, the first stage involves creation of data structure and the second one is about performing the retrieval. The key points of this design include:

- Before executing the query, the Search Engine learns about the Database Schema and creates an Internal data structure, which corresponds to this schema.
  There are two ways of implementing this idea. First, involves flooding all the data into the Internal data structure during the preprocessing stage. Then, all the future retrievals are only based on this Internal data structure. The role of this Internal data structure is similar to the role of Oracle tables in the previous proposal. It is a structured mapping of XML Database.
  In the second approach an Internal data structure is created but is not populated by the data. The search engine can create the Internal data structure only if it knows

the Database Schema. It does not need to touch the XML database in the process of creating the Internal data structure. When performing the retrieval, the search engine needs to parse the XML documents and extract the data item following the *XML Pointers*. We follow the first alternative.

- An XML parser for the Search Engine to follow the XML Pointers.
- The Creation of the Internal data structure can be split into two stages.

  a) *Create the Data Structure.* The Database Schema is enough to provide all the necessary information to construct the Internal data structure, and

  b) *Flooding the Internal data structure.* The Search Engine needs to parse the XML documents, extract the data and flood them into the Internal data structure.

Query Input

Depending on what kind of Query Interface the system provides to users, the queries may or may not reflect the Database Schema. Therefore our design of Search Engine includes query translation.

Retrieval based on the Internal Data Structure

When the Internal data structure is flooded with the data, then the retrieval is rather simple. The IR system only requires the design of a group of operations on the Internal data structure, and the retrieval is then performed using these primitives. Otherwise, the retrieval needs to parse the XML documents again, translate the format of the data, check if the data match the query criteria, etc.

Buffering and Communicating the Retrieval Results to Query Interface

Our design implements internal buffers to hold the retrieval results. When the retrieval process is complete the results are communicated to the Query Interface. Because the Search Engine and the Query Interface are separately developed, a protocol is needed to make sure they can communicate effectively.

Displaying the Retrieval Results

The results of the retrieval process are expressed in HTML format and hence can be displayed in any Web Browser. Since the original format of the data is an XML format, we need a format translation from the XML to HTML.

In this subsection we describe the individual components of the XML-based IR system and their functionality.

1. **Web Crawler:** A Web Crawler is designed to enhance the performance of the the system. A seed URL of a Web site is input to the Web Crawler. As it explores a site,the Web Crawler downloads only those pages, the URL domains of which matches with that of seed URL. As a result, a local mapping of the remote site is built contains downloaded HTML data for conversion to XML/RDF. This component could also be used to make a copy of a remote site so that information searching is efficient.

2. **Translator:** A translator converts the downloaded HTML data to an XML/RDF. Conceptually, the input to the Translator is the local mapping of a remote site and an RDF schema. For the conversion to actually take place, this component employs a home-grown rule called Simplified XML (SXML) to help guide its conversion process. Specifically, SXML dictates rules that instruct the Translator where, in HMTL pages, to insert the appropriate XML/RDF metadata (defined in the RDF schema). SXML is an object-oriented rule designed for simple conversion of HTML data to XML/RDF. In our implementation, the Translator only works for Web pages whose primary display elements are HTML tables and lists.

   The translation from HTML to XML documents includes the generation of XML tags and insertion of these XML tags into the HTML documents. The metatags are created from metadata describing web resources and encoding them following the Resource Description Format Schema and Syntax specification. The Translator accepts these metatags, which are similar to XML tags, and expand these tags to XML tags one by one. Following each expansion, the insertion is performed. The process of scanning the metatags consists of alternating sequence of expansion and insertion until all XML tags have been generated and inserted in the document. The metatags contain two kinds of information – the type of XML tags generated and the place for inserting tags. The insertion of tags into HTML documents is a fairly complex process and needs to be customized for each site.

3. **Search Engine:** The search engine provides precision searches based on descriptive XML/RDF metadata. This ability is enabled by an XML/RDF database built by Search Engine corresponding to the local mapping of a site. Our implementation of the Search Engine supports querying only using Boolean operators. To support interchange of information, i.e. combining of search results, among Web sites the Search Engine also encodes its results in XML/RDF. In other words, the component takes advantage of RDF's interoperability feature to make its search results usable outside of the local search process.

   The key function of Search Engine is to create the internal data structure to hold all related information generated by *X-Pointers* in XML documents. The Search Engine scans the entire local XML document hierarchy to retrieve all the relevant data from the documents and put them into the internal data structure. In this project we adopt a simplified implementation. In a commercial implementation, the design of a search engine may attain high efficiency using memory and disk cache ability. Therefore,

a search engine may create the *Index Internal Database* within the initial scan of the XML document hierarchy which is known as the *Indexing* process in some IR systems. The *Index Database* contains all the indexing information of the documents. It works like a "*lookup table*". When a query arrives, the Search Engine looks up the "table", determines which XML documents should be analyzed and then accesses the files directly. This scheme does not demand huge storage to hold all the data. The Index Database can adopt "*inverted table*" or implement another structure.

When the first scan is finished, the *Internal Database* is created, and the retrieval is performed via a group of standard "*primitives*". The *primitives* are a group of functions. They act as a *Data Manipulation Language* in Database Management Systems. All the *primitives* are *Read-only* and based on the *Internal Data Structure.* Actually, the Search Engine and Query Interface are incorporated into one entity, though they belong to separate functional units if we make the design more formal.

This design is a simplified version of the system. It follows the *X-Pointers* to trace the available information transported in the XML documents and apply the database retrieval technology to IR.

4. **Negotiator:** The function of a negotiator is provide access to multiple Web sites and offers the user the ability to combine search results obtained from various sites. It is implemented using a layered architecture and access to individual sites is facilitated using an application server. Java RMI is used to implement the server because it can simulate distributed query processing and at the same time does not require each Search Engine to have its own server for receiving queries. When the Negotiator receives a query from the user it analyses the query and decomposes it into fragments and sends them to appropriate sites for execution. Once the query fragments return, search results are combined automatically and sent to the user via the User Interface component. The processing of user queries is accomplished in Negotiator using two sub-components: Query Processor and XML/RDF Parser. The former is responsible for retrieving the information needed from each site and post-processing (or, combining) them, if specified in the query. The latter is internally used by Negotiator to parse individual search results into data structures that can be manipulated by Query Processor. Negotiator supports querying and combining of information using an SQL-like query language with three operations allowed: SELECT, PROJECT, and JOIN. The rationale behind this choice is that XML/RDF data is semi-structured in nature and benefits from structured query languages. Joining is possible because search results are represented in an interoperable format, RDF, which also allows for checking of consistency of metadata about semantics, interrelationship constraints, and so on. Negotiator also wraps its results in XML/RDF so that, if scalability is an application's goal then it can be achieved by considering Negotiator as a single search engine. Thereby, increasing the number of sites that can be queried without introducing too much complexity into the application.

5. **Converter:** A converter enables incorporation of more than one Web site into the system without requiring the individual systems to have the same searching/querying capabilities as those of Negotiator. As such, Converter's primary responsibility is to convert a query fragment received from Negotiator to the native query language used by the search engine for which Converter does the conversion.

6. **User Interface:** The User Interface is implemented using Web browsers such as Netscape or Internet Explorer for wide availability. The interface consists of a query window, where users can specify their information needs, and a result screen, which pops up as a result of the users submitting their query. Though encoded in XML/RDF search results are displayed using HTML, via a simple mapping from XML/RDF, to make use of HTML's powerful display capabilities.

All components of the system were implemented using Java, Javascript, XML/RDF, and HTML.

2.1.1 Experimental Results

The following are the important results of IR system implementation:

- Use of XML/RDF (semantics and logical structures of data) yields improved precision and recall measures: 100% for precision, approximately 70% for recall based on data obtained from 2 existing Web sites:

  www.edmunds.com and
  www.highwaysafety.org/vehicle_ratings/ratings.htm

  and sample queries.

- Encoded in XML/RDF, query results from different information sources (Web sites) can be combined. Logical meaning and usage of XML/RDF metadata are resolved with RDF schemas, which serve as reference points for querying application during processing.

- Similarity between RDF triples (resource, property, value) and relational records simplifies information combining and querying. In other words, the problem is reduced to record matching. The mapping can be utilized in future developments to store larger collections of XML/RDF data in RDBMS's for more efficient processing. The mapping is as follows:

  - A record is an RDF Subject (a resource)
  - A field is an RDF Property
  - A record field is an RDF Value

- XML/RDF data are semi-structured in nature and therefore can benefit from structured query languages. The simple query language used in the project has the following form and allows SELECT, PROJECT, and JOIN operations on data: SELECT attributes/tags FROM sources WHERE conditions

- Existing HTML data can be converted to XML/RDF with the home-grown development of HTML-to-XML/RDF conversion rule: Simplified XML (SXML). Currently, application of the rule only applies to HTML table data. SXML supports object-oriented conversion of data and the following rules are available: PREDICATE, PROPERTY, OBJECT, and VALUE.

- With 3-tier system architecture (client/application service/data sources) and the export of XML/RDF-encoded data, information sources can be considered as one unified source so that it can be incorporated to form larger systems (future development).

- Platform independence with use of Java as implementation language and Web browsers such Netscape, Internet Explorer

- Experimental results, i.e. improved precision and recall as well as interoperability among the sources, prove the XML/RDF approach to automated processing of metadata.

2.2 Minimal Term Sets

There is considerable interest in bridging the terminological gap that exists between the way users prefer to specify their information needs and the way queries are expressed in terms of keywords or text expressions that occur in documents. One of the approaches proposed for bridging this gap is based on technologies for expert systems. The central idea of such an approach was introduced in the context of a system called Rule Based Information Retrieval by Computer (RUBRIC)[1]. In RUBRIC, user query topics (or *concepts*) are captured in a rule base represented by an AND/OR tree. The evaluation of AND/OR tree is essentially based on minimum and maximum weights of query terms for conjunctions and disjunctions, respectively. The time to generate the retrieval output of AND/OR tree for a given query topic is exponential in m, where m is the maximum number of conjunctions in the DNF expression associated with the query topic.

We propose a new approach for computing the retrieval output. The new approach involves preprocessing of the rule base to generate Minimal Term Sets (MTSs) that speeds up the retrieval process [8*][1]. The computational complexity of the on-line query evaluation following the preprocessing is polynomial in m. We show that the computation of MTSs allow a user to choose query topics that best suit their needs and to use retrieval functions that yield a more refined and controlled retrieval output than is possible with the AND/OR tree when document terms are binary.

[1] A cite with an asterisk is found in List of Publications in Section 3.

2.2.1 Enhanced Minimal Term Sets

We develop strategies for enhancing retrieval output by incorporating document term weights into computation of document-query similarity measure for both AND/OR Goal tree as well as MTS, based on p-Norm model [4].

2.2.1.1 AND/OR Goal Trees. The evaluation of AND/OR Goal tree is a rule-based retrieval model, for a given query topic, based on recursive application of the following calculus.

1. Minimum query term weights in a conjunction.
2. Maximum query term weights in a disjunction.

In addition, for evaluating an abstract query term, the following additional calculus is needed.

3. Product of a user defined weight and the propagated value obtained by Steps 1 and 2.

In this rule-based retrieval model, document term weights are assumed to be binary. One problem with this kind of computation calculus is to rank documents depending only on the lowest or highest weights of query terms for conjunctions and disjunctions, respectively, which suffers, unfortunately, from a lack of discrimination among documents nearly to the same extent as the conventional retrieval systems do.

2.2.1.2 Incorporating Document Term Weights into AND/OR Goal Tree. Here, we investigate the case of AND/OR Goal trees where document term weights are incorporated into the computation of similarity between a query and a document [6*]. In this case, we need to find a finer computation of retrieval status values other than that of the rule-based retrieval model. We consider Extended Boolean Retrieval model developed by Salton et al. [4] as our computational model and showed how query-document similarity was computed.

Two operators are introduced reflecting the extent of query-to-document similarity in an n-dimensional vector space. These operators are extensions to the generalized AND and OR queries. We call these operators AND_p and OR_p for the generalized AND and OR queries respectively.

Our strategy to incorporate document term weights into an AND/OR Goal tree is similar to the evaluation of AND/OR Goal Trees except the following:

> Consider a query in the form of AND/OR goal tree. We define an evaluation scheme for such a query as follows:
>
> 1. Instead of the operator min, apply AND_p formula for the conjunctions, and
> 2. Instead of the operator max, apply OR_p formula for the disjunctions.

We assume that a query topic is converted into its DNF. This not only simplifies the representation of the dependency relation between rules but also makes the AND_p and OR_p formulas simple to implement.

2.2.1.3 Incorporating Document Term Weights into MTS.

We develop a strategy for incorporating document term weights into MTS [6*]. Clearly, when documents with term weights are considered, the MTS generation discussed earlier is no longer valid due to the fact that the weights of conjunctions are computed under the assumption that the corresponding query terms are present in the documents, and then this assumption is verified during retrieval of documents. Furthermore, contrary to the case for AND/OR goal tree, we do not take document term weights into account when MTS for a given query topic is generated. Therefore, the disjunction of all conjunctions must be evaluated to rank documents.

Under the restrictions given above, we define an evaluation scheme based on AND_p and OR_p formulas for the MTS of a query topic as follows:

1. Assume that a query topic is converted to its MTS representation (i.e., disjunction of conjunctions of concrete query terms).

2. Consider the weight of a conjunction obtained as a result of MTS generation, as the extent of its relative importance in the query topic.

3. Preserve each concrete query term weight and apply AND_p formula to the corresponding conjunction.

4. Apply OR_p formula over conjunctions to get retrieval status value of a document for the query topic.

We develop an algorithm for computing query-document similarity between a document D and an MTS query Q using AND_p and OR_p operators. Here, first, the distance between each conjunction and the document D is computed. This distance is interpreted as the RSV of the document D when matched against a particular conjunction. After all these 'computed document weights' are determined, the relative distance of the document to the query topic is computed using OR_p formula. Finally, the result of OR_p will be treated as the retrieval status value of D.

2.2.2 On Modeling of Concept Based Retrieval in Generalized Vector Spaces

In RUBRIC [1], the retrieval output is determined by Boolean evaluation of the AND/OR tree, and can be enhanced by generating MTSs (discussed above.) However, since the Boolean evaluation ignores the term-term association unless it is explicitly represented in the tree, the terminological gap between users' queries and their information needs may still remains. To solve this problem, we adopt the generalized vector space model in which the term-term association was well established, and thus extending the RUBRIC model.

The basic idea of a GVSM is to specify a vector space with a set of vectors as its basis (these vectors are called the atomic expressions for the minterms and they are supposed to be mutually linearly independent). Each term and thus each document can be represented as a vector in the space. In addition, a query corresponding to the user's preference is also represented as a vector. By computing the similarity between the query vector and the

document vector, the RSV for a document corresponding to the query is obtained. The main advantage of GVSM is that the term-term relationships are also accounted for, therefore, it can give a fairly accurate retrieval output even if some terms in the query do not appear explicitly in a document. These term-term associations are computed as an integral part of the automatic indexing process. However, the difficulty of finding a query that is an accurate description of a user's need and easy to understand is still a challenge for GVSM. Therefore, we propose a strategy to integrate the ideas of concept based retrieval in RUBRIC with GVSM [2*].

2.2.2.1 Experimental Evaluation. We describe experiments to validate our approach. We want to show that our approach produced more appropriate ranking as compared to RUBRIC approach. We choose the *Google* search engine [5]. The user interface of Google is simple and it uses a conjunction of keywords as queries and returns a set of links to web pages. We computed 12 weighted MTSs and constructed a query for each MTS. For each query, we chose the top 20 links retrieved by Google. Since some links to the web pages might have been changed or moved after being indexed by Google, we must eliminate these links. Finally, we got a collection of 196 documents. The relevance judgments, for evaluation purposes, are determined by looking through each document and choosing those document related to the concept (query at hand. From the collected documents, we could construct the document-term matrix. Using the matrix, we could compute the term vectors and document vectors. With respect to each MTS, RSV was computed for each document. Further, these RSVs are combined into a single RSV for each document satisfying more than one MTS. As a result, the user could be given the option of performing the disjunction operation over some or all the MTSs. Thus, our approach for ranking required post-processing of results from Google.

In our experiment, we select all the MTSs and compute the performance in terms of recall-precision. We conduct experiments using two different disjunction operators, namely the standard disjunction operator '+' and our disjunction operator '$\oplus$'. We call the former *Extended Rubric version 1* (ER_1) and the latter *Extended Rubric version 2* (ER_2).

In order to compare earlier RUBRIC approaches, we conducted an experiment using two versions of the original RUBRIC model, denoted by R_1 and R_2, respectively. In R_1, we adopted the original idea of RUBRIC. That is, we use max operator and min operator for disjunction and conjunction, respectively. In R_2, we used the same operator (that is, min operator) for conjunction and used weighted operator for disjunction [1].

¿From the experiments, we summarize the following conclusions.

1. The original RUBRIC approach (R_1) gave inaccurate RSVs.
2. The variation of RUBRIC (R_2) did not improve the accuracy of RSVs.
3. The extended approaches (ER_1 and ER_2) showed a better performance than the original RUBRIC approaches (R_1 and R_2).
4. ER_2 showed a better performance than ER_1.

2.3 Optimal Queries in Information Filtering

Information filtering (IF) and Information Retrieval are "*information seeking*" type — which is a term that describes any process by which users can obtain information from automated information systems. Information filtering differs from Information Retrieval in the sense that the latter accepts user needs (requests) as a query and provide the user with a list of documents ordered (ranked) based on the similarity with the user query. Information filtering systems, on the other hand, monitors documents as they enter the system (newly arrived documents) and selects only the ones that match the user query (known as a *profile* in Selective Dissemination of Information [2]). Thus, IF seeks relevant/non-relevant decision rather than a ranked output as in IR [3]. As a result, the objective of an information filtering is to classify/categorize documents as they arrive into the system.

We investigate an information filtering method based on Steepest Descent Induction Algorithm combined with a two-level preference relation on user ranking [1*]. The performance of the proposed algorithm was experimentally evaluated. The experiments are conducted using Reuters-21578 and TREC-9 data collections.

2.3.1 Experiment Results on Reuter-21578 data collection

In this experiment, the performance of the proposed algorithm is experimentally evaluated. We investigated the system performance under various settings of parameters. A microaverage breakeven effectiveness measure was used for performance evaluation. The best size of negative data employed in the training set was empirically determined and the effect of R_{norm} factor on the learning process was evaluated. Finally, we demonstrated effectiveness of proposed method by comparing experimental results to other inductive methods. The following is a brief description of the Reuter-21578 data collections followed by the experiment results.

We used Reuters-21578 text categorization test collection Distribution 1.0 as our corpus [6]. This collection consists of 21,578 documents which are Reuters newswire stories. The documents of this collection are divided into training and test sets. We used the Modified Apte split that has 9,603 training documents, 3,299 test documents, and 8,676 unused documents. Furthermore, the training set was reduced to 7,775 documents as a result of screening out training documents with no assigned topic.

The results of the experiment is summarized as follows: The size and quality of the training set is an important issue in generating induction rules. In our preliminary experiments on Reuters-21578 dataset, we fond that as the proportion of negative to positive data is 50% or 80% the quality of induction is maximum. Also, in a another preliminary experiment on R_{norm} factor and its effect on the system performance, we observed that as R_{norm} was decreased the quality of the induced rule was decreased and hence the performance degraded. The average precision/recall of the system performance was computed over all 118 topics. The breakeven point is 81.28%. As SDA was compared to other inductive algorithms experimented on Reuters-21578, it outperformed Findsim, NBayes, and BayesNet methods. However, it competed with Decision Tree methods and outperformed by the Linear SVM

method.

2.3.2 Experiment Results on TREC-9 data collection

In this experiment, the performance of the proposed algorithm is also experimentally evaluated using TREC-9 data collection. The documents for the experiment were obtained from the 9^{th} Text REtrieval Conference (TREC-9)[2]. The collection is based on Financial Times news stories from 1991 to 1994. There were a total of 64139 documents. According to TREC-9 recommendations data set was divided into two groups - training and testing. The documents from the year 1992 were used in training session while those from 1993 to 1994 were used in testing. Each document was formatted to have XML specification. A total of 50 topics numbered from 351 to 400 were also provided by TREC-9 with the documents. These topics have been examined against the given document set and the relevance judgments for each topic were provided as part of the corpus.

In this experiment, after training the queries for each of the topics, the queries were evaluated for filtering decisions on test set of the corpus. Based on the ranking of the documents by the query, precision and recall values were calculated for different thresholds. A single composite precision Vs recall graph reflecting the average performance of all the queries in the system was reported. The breakeven for the system is approximately 41.12%.

The performance of SDA largely depends on the availability of relevant documents for a given topic to obtain an optimal query. In our case, the availability of such positive examples for some of the queries remained an outstanding issue as the number of relevant documents provided were too small or did not exist at all. This is reflected in the low breakeven points obtained for some of the queries. On the other hand, the breakeven points for some of the queries are shooting as high as 66%. Apparently, it shows a wide gap between system performance but actually both trends are in accord with the philosophy around which SDA revolves — the more positive examples for query training, the more precise the query is and vice versa.

2.4 Formal Concept Analysis

2.4.1 Definable Concepts

The existing notion of concept, which we call atomic concept, allows only for conjunctions in its extent and intent. We generalize the notion of atomic concepts and introduced definable concepts [21*]. Definable concepts allow for disjunctions in addition to conjunctions in their definitions. Furthermore, an atomic concept is a special case of a definable concept.

2.4.2 Concept Approximation

Atomic and definable concepts have precise mathematical definitions. However, for many applications, such as information retrieval, the concepts we deal with do not fulfill these

[2]http://trec.nist.gov/

definitions and therefore can not be described as atomic or definable concepts. To deal with these concepts and any concepts that can not be classified as definable concepts (note that an atomic concept is also a definable concept), we introduce the notion of non-definable concepts [21*]. We give classifications to many cases one can have a non-definable concepts.

We introduce the notion of concept approximation to mean finding definable concept(s) that best describe a non-definable concept. We show how a non-definable concept can be approximated by atomic concepts [3*]. We use two different approaches for concept approximation. For the first approach, we use rough set theory [9*]. For the second approach, we use fuzzy set theory [5*]. We also use rough set theory to approximate non-definable concepts in terms of definable concepts [22*].

2.4.3 Using Closed Itemsets for discovering representative association rules

We use the ideas of formal concept analysis for an application in data mining [4*]. A set of representative association rules is a minimal set of rules that covers all the association rules and from which all association rules can be generated.

We use the notion of an atomic concept to describe and represent a closed itemset. Frequent closed itemsets correspond to the extents of the atomic concepts in the given database that satisfy a minimum support constraint. The minimum support constraint is used to distinguish the interesting association rules from the uninteresting ones. We proved that it is sufficient to only use frequent closed itemsets instead of all the frequent itemsets for finding all the association rules.

Using frequent closed itemsets for discovering representative association rules results in big reduction in the input size for the algorithms used to find representative association rules.

3 List of Publications and technical reports

The papers published and manuscripts submitted are listed below.

a. Papers published

1. "Optimal Queries in Information Filtering", A. Alsaffar, J. S. Deogun, and H. Sever. In *Foundations of Intelligent Systems: Twelfth International Symposium on Methodologies for Intelligent Systems, ISMIS'2000 proceedings*, Zbigniew Ras, and Setsuo Ohsuga (eds.), Lecture Notes in Artificial Intelligence, Vol. 1932, pp. 435–443, Springer-Verlag, Charlotte, NC, October, 2000.

2. "On Modeling of Concept Based Retrieval in Generalized Vector Spaces", M. Kim, A. Alsaffar, J. S. Deogun, and V. V. Raghavan. In *Foundations of Intelligent Systems: Twelfth International Symposium on Methodologies for Intelligent Systems, ISMIS'2000 proceedings*, Zbigniew Ras, and Setsuo Ohsuga (eds.), Lecture Notes in Ar-

tificial Intelligence, Vol. 1932, pp. 453–462, Springer-Verlag, Charlotte, NC, October, 2000.

3. "Concept Approximations for Formal Concept Analysis", J. Saquer and J. Deogun. To appear in Proc. of *The 8^{th} International Conference on Conceptual Structures (ICCS'2000)*, Darmstadt, Germany, August 2000, accepted.

4. "Using Closed Itemsets for Discovering Representative Association Rules", J. Saquer and J. Deogun. In *Foundations of Intelligent Systems: Twelfth International Symposium on Methodologies for Intelligent Systems, ISMIS'2000 proceedings*, Zbigniew Ras, and Setsuo Ohsuga (eds.), Lecture Notes in Artificial Intelligence, Vol. 1932, pp. 495–504, Springer-Verlag, Charlotte, NC, October, 2000.

5. "A Fuzzy Approach for Approximating Concepts", J. Saquer and J. Deogun. In *RSCTC'2000*, Banff, Canada, Oct. 2000, accepted.

6. "Enhancing Concept-based Retrieval based on Minimal Term Sets," A. Alsaffar, J. S. Deogun, V. V. Raghavan, and H. Sever. *Journal for Intelligent Information Systems*, Vol. 14, No. 2/3, pp. 155-174, 2000.

7. "Consecutive Retrieval Property — Revisited," J. S. Deogun and K. Gopalakrishanan. *Information Processing Letters*, Vol. 69, pp. 15-20, 1999.

8. "Concept-based Retrieval with Minimal Term Sets," A. H. Alsaffar, J. S. Deogun, V. V. Raghavan, and H. Sever. *Foundations of Intelligent Systems: Eleventh International Symposium, ISMIS'99*, Zbigniew Ras, and Andrzej Skowron (eds.), Lecture Notes in Artificial Intelligence, Vol. 1609, pages 114–122. Springer-Verlag, Warsaw, Poland, Jun. 1999.

9. "Formal Rough Concept Analysis," J. Saquer and J. S. Deogun. *Proc. of 7th International Workshop, RSFDGrC'99*, Yamaguchi, Japan, November 9-11, 1999. *New Directions in Rough Sets, Data Mining, and Granular-soft Computing*, Zhong, N., Skowron, A., and Ohsuga, S., (eds.), Lecture Notes in Computer Science, Vol. 1711, Springer-Verlag, 1999, pp. 91-99.

10. "The Item-set Tree: A Data Structure for Data Mining," A. Hafez, V. V. Raghavan, and J. S. Deogun. *Proc. of Data Warehousing and Knowledge Discovery (DaWaK'99) Conf.*, Florence, Italy, Aug. 99, 1999, pp. 136-145.

11. "Conceptual Clustering in Information Retrieval," S. K. Bhatia and J. S. Deogun. IEEE *Transactions on Systems, Man & Cybernetics*, Vol. 28, No. 3, pp. 427-436, 1998.

12. "Association mining and formal concept analysis," J. S. Deogun, V. V. Raghavan, and H. Sever. *Proc. of RSDMGrC98- Sixth International Workshop on Rough Sets, Data Mining and Granular Computing*, Research Triangle Park, NC, Oct. 1998.

13. "Structural abstractions of hypertext documents for web-based retrieval," J. S. Deogun, V. V. Raghavan, and H. Sever. *Proc. of DEXA 98 - 9th International Workshop on Database and Expert Systems Applications*, pages 385–390, Vienna, Austria, Aug. 1998.

14. "On Feature Selection and Effective Classifiers," J. S. Deogun, S. K. Choubey, V. V. Raghavan, and H. Sever. *Journal of Amer. Soc. for Information Sci.*, Vol. 49(4), pp. 423-434, April 1998.

15. "Data mining: Trends and issues-guest editors' introduction," V. V. Raghavan, J. S. Deogun, and H. Sever. *Journal of Amer. Soc. for Information Sci.*, Vol. 49(4), pp. 397-402, April 1998.

16. "Algorithms for the Boundary Selection Problem in Information Retrieval," J. N. Bhuyan, J. S. Deogun, and V. V. Raghavan. *Algorithmica*, Vol. 17, pp. 133-161, January, 1997.

17. "A comparison of classification methods," H. Sever, V. V. Raghavan, J. S. Deogun, and S. K. Choubey. *Proc. of International Conf. of Information Sciences- Rough Set & Computer Science*, volume 3, pages 371–374, Raleigh, NC, Mar. 1997.

18. "A comparison of feature selection algorithms in the context of rough classifiers," S. K. Choubey, J. S. Deogun, V. V. Raghavan, and H. Sever. *Proc. of International Conference on Fuzzy System (IEEE-Fuzz' 96)*, New Orleans, LA., Sept. 96, Vol. 2, pp. 1122-1128.

19. "Conceptual Query Formulation and Retrieval," S. K. Bhatia, J. S. Deogun, and V. V. Raghavan. *Journal of Intelligent Information Syatems*, Vol. 5, No. 3, pp. 183-209, November 1995.

20. "Exploiting upper approximations in the rough set methodology," J. S. Deogun, V. V. Raghavan, and H. Sever. *Proc. of the First International Conference on Knowledge Discovery and Data Mining,* Montreal, Canada, August 1995, pp. 69-74.

b. Papers submitted

21. "Formal Concept Analysis: Generalizations and Approximations", J. Saquer and J. Deogun. Submitted for publication in *Journal of Discrete Applied Mathematics.*

22. "Concept Approximations Based on Rough Sets and Similarity Measures", J. Saquer and J. Deogun. "Submitted for publications in *The International Journal of Applied Mathematics and Computer Science*, special issue on rough sets and their applications.

4 Scientific Personnel

Following is a list of scientific personnel supported by the grant.

1. Dr. Jitender Deogun, PI.
2. Dr. Ashok Samal, Co-PI.
3. Dr. Sharad C. Seth, Co-PI.
4. Amartya Sarkar, M.S.
5. Hui Zhang, M.S., Ph.D.(did not complete.)
6. Jamil Saquer, Ph.D.
7. Xiaoming Sun, Ph.D.(did not complete.)
8. Ali Alsaffar, Ph.D.
9. Salman Khan, M.S.
10. Tung Le, M.S.

References

[1] B.P. McCune and R.M Tong and J.S. Dean and D.G. Shapiro, "RUBRIC: A system for rule-based information retrieval", *IEEE Trans. on Software Engineering*, 11(9): 939–944, 1985.

[2] G. Kowalski, "Information Retrieval Systems. Theory and Implementation", Kluwer Academic, Norwell, Massachusetts, 1997.

[3] N.J. Belkin and W.B. Croft, "Information Filtering and Information Retrieval: Two sides of the same coin?", *Communications of the ACM*, 35(12):29–38, 1992.

[4] G. Salton and E.A. Fox and H. Wu, "Extended Boolean Information Retrieval", *Communications of the ACM*, 26(11):1022–1036, 1983.

[5] Google Search Engine, http://www.google.com.

[6] Reuters-21578 collection, http://www.research.att.com/~lewis